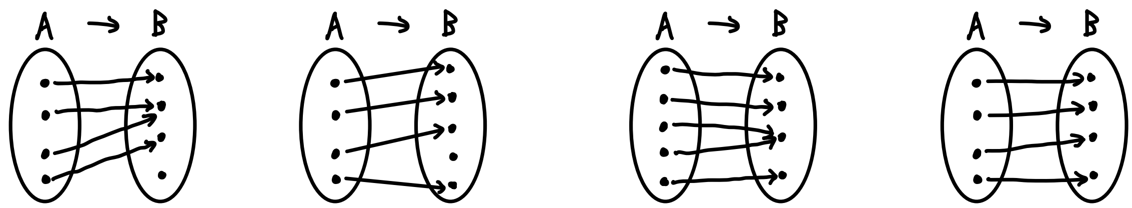
Announcements

- Regrade Requests open, due 3/16
- Clobber Policy: 50% both ways
- One-on-ones: form due 3/14
- No-HW option form due 3/14

Countability



general function

injection
one-to-one
$f(x_1) = f(x_2) \Rightarrow x_1 = x_2$
$|A| \leq |B|$

surjection
onto
$(\forall y \in B)(\exists x \in A)(f(x) = y)$
$|A| \geq |B|$

bijection
one-to-one and onto
$|A| = |B|$

Examples of countable sets: $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{N} \times \mathbb{N}$, set of all finite bitstrings, any finite set

Examples of uncountable sets: $\mathbb{R}$, $[0, 1]$, $\mathcal{P}(\mathbb{N})$, set of infinite-length bitstrings

Computability

Halting Problem: $\text{TestHalt}(P, x) = \begin{cases} \text{"yes" if } P \text{ halts on input } x \\ \text{"no" if } P \text{ does not halt on input } x \end{cases}$

Contradictory Example:

```
Turing(P):
    if TestHalt(P, P) = "yes":
        loop forever
    else:
        halt
```

Then Turing(Turing) is a contradiction!

CS 70 Discrete Mathematics and Probability Theory
Spring 2022 Koushik Sen and Satish Rao

DIS 7A

1 Countability: True or False

(a) The set of all irrational numbers $\mathbb{R}\backslash\mathbb{Q}$ (i.e. real numbers that are not rational) is uncountable.

(b) The set of integers x that solve the equation $3x \equiv 2 \pmod{10}$ is countably infinite.

(c) The set of real solutions for the equation $x + y = 1$ is countable.

For any two functions $f : Y \to Z$ and $g : X \to Y$, let their composition $f \circ g : X \to Z$ be given by $f \circ g = f(g(x))$ for all $x \in X$. Determine if the following statements are true or false.

(d) f and g are injective (one-to-one) $\implies$ $f \circ g$ is injective (one-to-one).

(e) f is surjective (onto) $\implies$ $f \circ g$ is surjective (onto).

a) True, since $\mathbb{Q}$ is countable and $\mathbb{R}$ is uncountable

b) True, since integers are countable

c) False, set of $[0,1]$ is uncountable

d) True; if $(f\circ g)(x_1) = (f\circ g)(x_2)$ then $f(g(x_1)) = f(g(x_2)) \Rightarrow g(x_1) = g(x_2) \Rightarrow x_1 = x_2$

e) False; consider $f(x) = x$ and $g(x) = x^2$ where $f, g : \mathbb{R} \to \mathbb{R}$.
Then there is no $x \in \mathbb{R}$ s.t. $f(g(x)) = -1$.

2 Counting Cartesian Products

For two sets A and B, define the cartesian product as $A \times B = \{(a,b) : a \in A, b \in B\}$.

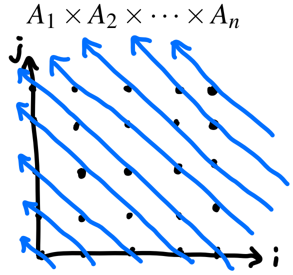
(a) Given two countable sets A and B, prove that $A \times B$ is countable.

(b) Given a finite number of countable sets $A_1, A_2, \ldots, A_n$, prove that

$$A_1 \times A_2 \times \cdots \times A_n$$

is countable.

a) $A = \{A_1, A_2, A_3, \ldots\}$
$B = \{B_1, B_2, B_3, \ldots\}$



Each coordinate represents a pair (A_i, B_j).

b) By induction on n.

Base Case: $n=1$. A_1 is countable.

Induction Hypothesis: $B = A_1 \times \ldots \times A_k$ is countable.

Inductive Step: $A_1 \times \ldots \times A_k \times A_{k+1} = B \times A_{k+1}$ is countable by part (a).

3 Undecided?

Let us think of a computer as a machine which can be in any of n states $\{s_1, \ldots, s_n\}$. The state of a 10 bit computer might for instance be specified by a bit string of length 10, making for a total of 2^{10} states that this computer could be in at any given point in time. An algorithm $\mathscr{A}$ then is a list of k instructions $(i_0, i_1, \ldots, i_{k-1})$, where each i_ℓ is a function of a state c that returns another state u and a number j describing the next instruction to be run. Executing $\mathscr{A}(x)$ means computing

$$(c_1, j_1) = i_0(x), \qquad (c_2, j_2) = i_{j_1}(c_1), \qquad (c_3, j_3) = i_{j_2}(c_2), \qquad \ldots$$

until $j_\ell \geq k$ for some ℓ, at which point the algorithm halts and returns $s_{\ell-1}$.

(a) How many iterations can an algorithm of k instructions perform on an n-state machine (at most) without repeating any computation?

(b) Show that if the algorithm is still running after $nk+1$ iterations, it will loop forever.

(c) Give an algorithm that decides whether an algorithm $\mathscr{A}$ halts on input x or not. Does your contruction contradict the undecidability of the halting problem?

a) k instructions, n possible states for each = nk

b) After nk+1 instructions, we have repeated a computation with the machine in the same state, so the same sequence of computations will be repeated.

c) Run A for nk+1 iterations. If it is still running, the program doesn't halt. This is not a contradiction because A needs to be run on an n-state machine, not any arbitrary algorithm.

4 Code Reachability

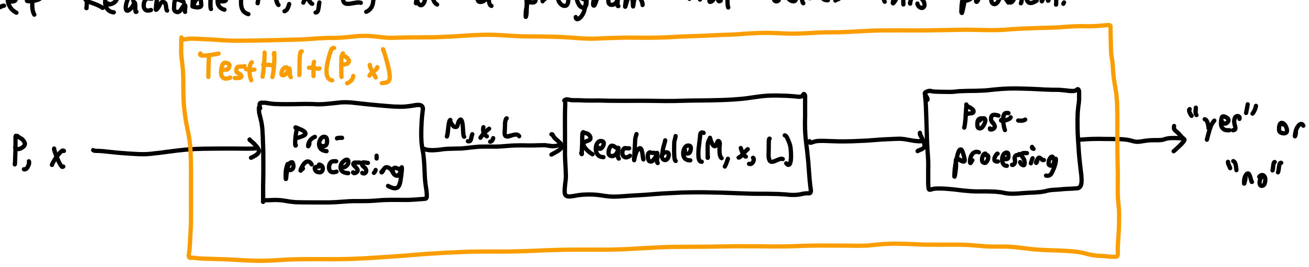
Consider triplets (M, x, L) where

```
M is a Java program
x is some input
L is an integer
```

and the question of: if we execute $M(x)$, do we ever hit line L?

Prove this problem is undecidable.

Let Reachable(M, x, L) be a program that solves this problem.




```
TestHalt(P, x):
    def M(t):
        P(t)
        return
    return Reachable(M, x, 2)
```